Homework\_Permutation

Meng-Hsin, Wu

2022/3/21

## Import dataset and check the data

########Import datasets################  
rm(list=ls())  
setwd("~/Courses of 110 semester-2/Intensive Statistical Ecology/0303/mock\_dataTS/mock\_dataTS")  
library(readxl)  
FishCopData<-read\_xls('enviANDdensity.xls')  
FishCopDens<-FishCopData[11:12]  
colnames(FishCopDens)<-c('Fish', 'Copepod')  
FishDens<-FishCopDens$Fish  
CopDens<-FishCopDens$Copepod  
print(FishCopDens)  
###########SP dataset####################  
setwd("~/Courses of 110 semester-2/Intensive Statistical Ecology/0303/mock\_dataTS/mock\_dataTS")  
library(readxl)  
CopData<-read\_xls('copepod\_datasheet.xls',  
 col\_names =T)  
CopData<-t(CopData)  
CopData<-as.data.frame(CopData)  
row.names(CopData)<-CopData$V1  
CopData<-CopData[-1,]  
colnames(CopData)<-CopData[1,]  
CopData<-CopData[-1,]  
  
#Confirm that all data in this data frame are characters  
CopSPData<-sapply(CopData[2:182], FUN=as.numeric)  
CopSPDensity<-CopSPData\*CopDens #Get the species density  
  
CopSPDensity<-as.data.frame(CopSPDensity)  
OVdensity<-CopSPDensity$`Oncaea venusta`  
CPdensity<-CopSPDensity$`Canthocalanus pauper`  
OVCP<-cbind(OV=OVdensity, CP=CPdensity)  
OVCP<-as.data.frame(OVCP)  
OVCP<-OVCP/100  
ov<-OVCP$OV  
cp<-OVCP$CP  
  
print(OVCP)

## Question\_1: compute coefficients between fish and copepod

#################lm function##################  
MylmCoefficient<-function(x,y){  
 is.array(x)  
 is.array(y)  
 beta1\_up<-sum((x-mean(x))\*(y-mean(y)))  
 beta1\_down<-sum((x-mean(x))^2)  
 beta1<-beta1\_up/beta1\_down  
 beta0<-mean(y)-beta1\*mean(x)  
   
 return(c(beta1, beta0))  
}  
###############################################  
paste("beta\_1: ",MylmCoefficient(CopDens, FishDens)[1], ", beta\_2: ",MylmCoefficient(CopDens, FishDens)[2])

## Question\_1: Generate null distribution

#######Functions Construction#################################  
mySampling<-function(n){  
 R<-c()  
 StudentLeft<-list()  
 StudentLeft[[1]]<-c(1:n)  
 StudentSelect<-c()  
 for(i in 1:n){  
 R[i]<-ceiling(runif(1, 0, length(StudentLeft[[i]])))   
 StudentLeft[[i+1]]<-StudentLeft[[i]][-R[i]]  
 StudentSelect[i]<-StudentLeft[[i]][R[i]]  
 }  
 StudentSelect[[n]]<-StudentLeft[[n]]  
 return(StudentSelect)  
}  
  
####################################  
  
myPermutation<-function(x, n=5000){  
 is.array(x)  
 permute<-as.data.frame(matrix(nrow=length(x), ncol=n))  
 for(i in 1:n){  
 sequence<-mySampling(length(x))  
 permute[i]<-x[sequence]  
 }  
 return(permute)  
}  
  
##################################  
  
myPermutPairs<-function(x, y, n=5000){  
 is.array(x)  
 is.array(y)  
 bind<-c(x,y)  
 permute<-as.data.frame(matrix(nrow=length(bind), ncol=n))  
   
 for(i in 1:n){  
 sequence<-mySampling(length(bind))  
 permute[i]<-bind[sequence]  
 }  
 PermuteX<-permute[1:length(x),]  
 PermuteY<-permute[(length(x)+1):length(bind), ]  
 # permute.xy<-as.data.frame(matrix(nrow=length(x),  
 #ncol=2\*n))  
 #for(i in 1:n){  
 # permute.xy[2\*i-1]<-PermuteX[i]  
 # permute.xy[2\*i]<-PermuteY[i]  
   
 #}  
   
Result<-list(X=PermuteX, Y=PermuteY)  
return(Result)  
}  
####################################################  
######Replace the function ecdf######  
myECDF<-function(x, N){  
 is.array(x)  
 is.numeric(N)  
 if(N>max(x)|N<min(x)){  
 paste('Please do not do something silly')  
 }else{  
 x.sort<-sort(x)  
 below<-length(x.sort[x.sort<=N])  
 percentile<-below/length(x.sort)  
 return(percentile)  
 }  
}  
#########################################  
########################################  
  
myPercentile<-function(x, prob){  
 is.array(x)  
 is.numeric(prob)  
 x.sort<-sort(x)  
 position<-length(x.sort)\*prob  
 if(is.integer(position)==TRUE){  
 value<-(x.sort[position]+x.sort[position+1])/2  
 }else if(is.integer(position)==FALSE){  
 position1<-floor(position)  
 value<-x.sort[position1]}  
 return(value)  
}  
##################################################  
###########Correlation p value####################  
########Correlation p value###################  
MyCorrP<-function(x, y, n=5000){  
yPermute<-myPermutation(y, n)  
CoeffRandom<-c()  
for(i in 1:5000){  
 CoeffRandom[i]<-MylmCoefficient(x, as.vector(as.matrix(yPermute[i])))[1]  
}  
Beta1<-MylmCoefficient(x,y)[1]  
if(Beta1>max(CoeffRandom)|Beta1<min(CoeffRandom)){  
 return(0)  
}else if(Beta1>median(CoeffRandom)){  
 Pvalue<-1-myECDF(CoeffRandom, Beta1)  
 return(Pvalue)  
}else if(Beta1<=median(CoeffRandom)){  
 Pvalue<-myECDF(CoeffRandom, Beta1)  
 return(Pvalue)  
}  
#Pvalue<-1-myECDF(CoeffRandom, Beta1)  
#return(Pvalue)  
}  
##################################################  
FishPermute2<-myPermutation(FishDens, 5000)  
CoeffRandom<-c()  
for(i in 1:5000){  
 CoeffRandom[i]<-MylmCoefficient(CopDens, as.vector(as.matrix(FishPermute2[i])))[1]  
}  
paste('Check the distribution of randomized beta\_1')  
hist(CoeffRandom, breaks=100,  
 main=expression(paste('Permutation of ', beta, '1')), xlab=expression(paste(beta, '1')), las=1)

## Question\_1: Compute P value and examine significance

PvalueBeta1<-MyCorrP(CopDens,  
 FishDens,  
 5000)  
  
paste('P value=', PvalueBeta1)  
paste('P<0.05')  
paste('Significant difference from null')

## Question\_2: Inspecting the significant difference between the density of two copepod species

####Question\_2############################  
ovcpPermute<-myPermutPairs(ov, cp, 5000)  
ovPermute<-ovcpPermute$X  
cpPermute<-ovcpPermute$Y  
ovMean<-c()  
cpMean<-c()  
for(i in 1:5000){  
 ovMean[i]<-mean(as.matrix(ovPermute[i]))  
 cpMean[i]<-mean(as.matrix(cpPermute[i]))  
}  
#################################  
Mysapply<-function(x, FUN='mean'){  
is.data.frame(x)  
 k<-c()  
 if(FUN=='mean'){  
 for(i in 1:ncol(x)){  
 k[i]<-mean(x[,i])  
 }  
 return(k)  
 }else if(FUN=='median'){  
 for(i in 1:ncol(x)){  
 k[i]<-median(x[,i])  
 }  
 return(k)  
 }else if(FUN=='sum'){  
 for(i in 1:ncol(x)){  
 k[i]<-sum(x[,i])  
 }  
 return(k)  
 }else if(FUN=='max'){  
 for(i in 1:ncol(x)){  
 k[i]<-max(x[,i])  
 }  
 return(k)  
 }else if(FUN=='min'){  
 for(i in 1:ncol(x)){  
 k[i]<-min(x[,i])  
 }  
 return(k)  
 }else if(FUN=='sd'){  
 for(i in 1:ncol(x)){  
 k[i]<-sd(x[,i])  
 }  
 return(k)  
 }else if(FUN=='quantile'){  
 Q<-as.data.frame(matrix(nrow=4, ncol=ncol(x)))  
 for(i in 1:ncol(x)){  
 Q[i]<-myPercentile(as.vector(x[,i]),  
 c( 0.25, 0.5, 0.75, 1))  
 }  
 row.names(Q)<-c('0.25', '0.5',  
 '0.75', '1')  
 return(Q)  
 }  
}  
   
quantile.test<-Mysapply(ovPermute, 'quantile')  
####################################################  
ovMean2<-Mysapply(ovPermute,'mean')  
cpMean2<-Mysapply(cpPermute, 'mean')  
  
ovcpDiff<-ovMean-cpMean  
hist(ovcpDiff, breaks=100, main='Distribution of difference of the mean (randomization)', xlab='Difference of the mean', las=1)  
ovcpDiff.origin<-mean(ov)-mean(cp)  
Pvalue.SPdiff<-1-myECDF(ovcpDiff, ovcpDiff.origin)  
  
paste('P value: ', Pvalue.SPdiff)  
paste('P>0.05, ', 'No significant difference')  
#p=0.1292  
#No significance  
#############################################